

CS161 HOMEWORK ASSIGNMENT 6

DUE DATE – 5:00 PM MARCH 30TH, 2012

The Sisco router company has heard of the programming accomplishments of this class of CS161 and has contacted us to build a router simulator for them in Java. Their current router simulator is written in FORTRAN, and the person that wrote it has retired and quit returning their calls. They see this piece as the final one to upgrade from the ancient technology. They have a significant amount of code already in place that calls the functions of the router, so their requirements for the simulator are very specific. In addition, they have provided an interface file based on their existing code. Your job is to write the simulator matching these requirements using Java.

Version 1.0 – Initial version

Version 1.1 – Clarified that the constructor of Router takes an int

Clarified that if no packets have been processed, the first packet is normal processing (i.e. no divide by zero)

If updates to this document are required, the version number will be updated appropriately. It is the student's responsibility to ensure his or her submitted document adheres to the latest version. Any changes to this document will be announced on the RamCT discussion board along with a description of the changes.

PURPOSE

The purpose of Homework Assignment 6 is to write a Java class using linked list technology that simulates the actions of a router. The basic function of the router is to take packets off the FIFO input queue and place them on the appropriate output queue to be transmitted on the wire. FIFO stands for First-In First-Out, meaning new packets are added to the tail of the queue, and packets to be processed are pulled from the head of the queue. Each output queue represents a port on router – the router's job is to ensure the packet is routed to the appropriate port. In addition, each router must handle service level responsibilities using linked list technology as well.

The concept of a service level is that packets of a certain type are guaranteed some percentage of processing. Let's use the example of high service level. If we say that packets with a service level of high (Let's call these SLH packets) are guaranteed to be at least SLHpercent% of the packets processed, then each time we go to retrieve a packet from the input queue, we must check whether that has occurred. If the number of SLH packets processed divided by the total packets processed (not counting the packet about to be processed) is less than SLHpercent%, we must look to see if there are any SLH packets to be processed. If so, the packet at the head of the SLH queue is selected and removed from both the SLH and the input queue. Once the service levels are satisfied, we would continue to process the packets off the head of the input queue on a first come first serve basis.

Note – since the input queue contains all packets of every service level priority in FIFO order, it is possible that SLH packets will be processed off the input queue. If so, these should be included in the number of the SLH packets processed.

Normally, the router would be event-driven (i.e. the arrival of a packet would trigger adding the packet to the queues), however since we are simulating this environment, all initialization parameters and commands will come from an input file specified on the command line.

REQUIREMENTS

You will write the Router class that implements IRouter and implement the stubbed out routines in SingleList and DoubleList. These routines are defined by their pre and post conditions. Under no circumstances should you modify the Packet.java , Assign6.java files, nor modify the SNode and DNode classes. If you modify these files and or classes, your program will not work with our test program and your grade will suffer.

You should write, debug, and test your SingleList and DoubleList classes first. Debugging the Router class without a thoroughly tested linked list class would be difficult. In addition – you can get points for these classes in case the Router class has issues.

INITIALIZATION

The Assign6 class will open the input file identified on the command line. If there is no file specified on the command line or the file specified on the command line does not exist, the Assign6 class should print the following error message and exit:

Usage: java RunRouter inputfile

The Assign6 class will read in the first number of the input file, **which is an integer** that will be the service level percentage for the service class high (i.e. 50 = 50%).

Once Assign6 has read in the service level number, it will create the Router class by calling the constructor and pass the service level numbers as arguments. Ex:

```
Router r = new Router (slhpercent);
```

During initialization, the Router class will create 2 doubly-linked lists and 2 singly-linked lists. These linked lists are:

- 1) The input queue (doubly-linked list)
- 2) The service level high queue (doubly-linked list)
- 3) The output queue for port 0 (singly-linked list)
- 4) The output queue for port 1 (singly-linked list)

The Router class will also need instance variables for total packets processed and SLH packets processed to be able to compute the statistics necessary to process the packets appropriately. Note that your doubly-linked list will use a head node, as discussed in class.

PROCESSING

Once the Router class has been created, the Assign6 class will continue to read commands from the input file and call the appropriate routines in the Router class to execute them.

These commands are listed below.

ADDPACKET

If the command is an ADDPACKET, Assign6 will create a packet from the information in the input file and pass it to Router to be added to the appropriate queues. When Router gets the packet, it will assign it a continuously increasing unique packet id, and place it on the appropriate linked lists. All packets are added to the tail of the input queue, and the message "Packet XXX was added to the input queue" (where XXX is the packet id) is printed to standard out. In addition, if the packet has a high service level, it is added to the tail of the appropriate queue and a message "Packet XXX was added to the service level high queue".

PRINT [OPTION]- where OPTION is required and is one of the following:

1. Starting at head, print the input queue in FIFO order.
2. Starting at head, print the service level high queue in FIFO order.
3. Starting at head, print the output queue for port 0 in FIFO order.
4. Starting at head, print the output queue for port 1 in FIFO order.

PROCESSPACKET

If the command is PROCESSPACKET, Assign6 simply calls the appropriate routine in Router. Router will:

- 1) Check to see if there are any packets on the input queue. If not, return.
- 2) Check to see if the percentage of SLH packets processed is greater than the SLH percentage.
 - a. If it is, perform the default processing
 - b. If the SLH queue is empty **or if this is the first packet to be processed**, perform the default processing
 - c. If both the above are not true, perform service level processing

Default processing

- 1) Remove a packet from the FIFO queue
 - a. If the packet is also on a service level queue, it must be removed from there as well
- 2) Check the port number in the packet
- 3) Add the packet to the appropriate output queue.
- 4) Print the following message to the console, filling in the # with appropriate values from the packet.

Processing packet #, adding to port #, with a service priority of #####.

Service level processing

- 1) Remove a packet from the appropriate service level queue
 - a. The packet must also be removed from the FIFO queue
- 2) Check the port number in the packet
- 3) Add the packet to the appropriate output queue.
- 4) Print the following message to the console, filling in the # with appropriate values from the packet.

Processing packet #, adding to port #, with a service priority of #####.

REMOVEPACKET

When Assign6 reads this command, it will read an additional integer from the input file. This integer is the packet id number of the packet to remove. Assign6 will then call the appropriate routine in Router to remove the packet with that id from the input queue as well as the service level queue, if it belongs to it. If the packet is removed, the following message(s) are printed to the console, filling in the # with the appropriate information. (Note: there will be 2 messages if the packet belonged to a service level queue as well).

Packet # was removed from the ##### queue.

WHAT TO TURN IN

You should turn the following files into RamCT:

1. SingleList.java
2. DoubleList.java
3. Router.java